NRZ Random Bit 동기를 위한 표본 위상 검출기

박세현[†] · 박세훈[††]

요 약

본 논문에서는 NRZ 랜덤 비트 동기를 위한 새로운 표본 위상 검출기(SPD: Sampling Phase Detector)를 제안한다. 제안하는 SPD는 국부 기준 신호의 주기와 입력 신호의 비트 구간의 위상 차의 평균값을 계산한다. 시뮬레이션과 실험 결과는 제안된 SPD가 NRZ 랜덤 신호의 Phase Detector로 유용하다는 것을 보여 준다. 제안된 SPD를 사용하여 NRZ 랜덤 비트 동기 회로를 설계하고 구현하였다.

Sampling Phase Detector for NRZ Random Bit Synchronization

Se-Hyun Park[†] and Se-Hoon Park[††]

ABSTRACT

This paper proposes a new type of Sampling Phase Detector (SPD) for NRZ random bit synchronization circuit. The proposed SPD calculates the mean value of phase difference between bit interval of input signal and period of local reference. Simulated and experimental results show that the proposed SPD is applicable to the phase detector for NRZ random signal. Finally the Random NRZ bit synchronization circuit. is designed and implemented by using SPD.

1. 서 론

멀티미디어 디지탈 통신에서는 전송의 대역폭을 줄이기 위해 NRZ(Non-Return to Zero) 형태의 신호를 많이 사용한다. NRZ 신호는 시간의 영역에서는 '0'과 '1'이 랜덤하게 존재하므로 클록주파수의 선스펙트럼을 가지지 않아 PLL(Phase Locked Loop)에 의한 추적에 어려움이 있다. 따라서 PLL을 사용하기 위해서 NRZ를 RZ(Return to Zero)로 바꾸어서 사용한다. NRZ를 RZ로 바꾸기 위해 비트 주기의 1/2배의 Delay을 주어 Excusive-OR를 하거나 단안정 회로를 사용한다. RZ는 NRZ에 비해 주엽(Mainlobe)의 대역폭이 2배에 가까우므로 전송에서는 자주 사용하지 않지만 각 비트 구간에는 반드시 천이를 가지므로 스펙트럼 상에서 클록주파수에 해당하는 선스펙트럼을 가지게 되어 PLL에 의한 추적이 가능하게 된다. 그러나 NRZ 신호를 RZ 신호로 바꾸기 위한 Delay회로나 단안정 회로의 비트 동기 방식은 주파수에 따라 설계 파라메터를 달리 해야 하므로 좋은 방법이라 할 수 없다[1,9].

본 논문에서는 NRZ 랜덤 비트 동기를 위한 SPD을 제안하고자 한다. SPD의 분류는 크게 4가지 유형으로 나누어 볼 수 있다[1,9]. 4가지 유형은 Flip-Flop SPD, Nyquist Rate SPD, Zero-Crossing SPD 및 Lead-Lag(LL) SPD로 구별된다. 제안하는 SPD는 LL SPD의 범주에 속할 수는 있지만 LL SPD의 동작원리와는 다르다. LL SPD는 국부기준에 대한 RZ 입력 신호의 sampling에 의해서 단순히 Lead-Lag의 2진법으로 동작한다. 그러나 제안된 SPD는 입력신호로 NRZ를 대상으로 할 수 있을 뿐 아니라 국부기준의 주기에 해당하는 윈도 내에서 입력 신호의 천이 지점의 변화에 의해 동작된다는 점에서 근본적 원리를 달리한다.

2. 관련 연구

디지털 VLSI의 발달과 함께 디지털 위상 동기 루

[†] 정회원, 미시건 주립대학 전기 컴퓨터 공학과의 겸임 교수
[††] 안동대학교 전자정보산업학부

프(DPLL:Digital Phase Locked Loop)는 기존의 아날로그 위상 동기 루프(APLL : Analog Phase Locked Loop) 회로를 디지털 회로로 대치한 장치로서 많은 장점을 가지고 있다. DPLL은 집적회로의 구현이 쉬울 뿐만 아니라 APLL에서 볼 수 있는 DC drift와 감도, 및 calibration등의 어려움을 해결할 수 있다. 다만 DPLL이 동작 속도의 한계와 양자화된 위상 해상도에 의해 위상 에러의 하한 값이 정해지는 단점이 있다.

DPLL이 가지는 3개의 하드웨어적 구성요소는 SPD, Digital Loop 필터 및 DCO(Digital Controlled Oscillator)이다. 4가지 유형의 SPD에 대해 간단히 살펴보자.

Flip-Flop SPD는 그림 1과 같이 Set-Clear F/F를 사용하여 F/F 출력에 의해 계수기를 동작시킨다. 계수기 출력은 위상 차에 비례하는 출력을 얻는다. 입력 v1에 대한 양의 영점 교차에서 F/F가 셋트되고, 국부추정 v2에서 F/F이 리셋된다.

Nyquist Rate SPD는 아날로그 신호를 대상으로 한다. 입력 신호를 균일 간격의 속도로 샘플링하여 샘플된 디지털 데이터는 DCO에서 발생된 신호와 Digital multiply를 한다. 따라서 Nyquist Rate SPD는 linear PD와 비슷하게 동작한다. 샘플링 주파수 f_s는 나이퀴스트 저역 통과 샘플링 이론을 적용한다. 따라서 입력 신호가 주파수 f_0인 사인파 신호
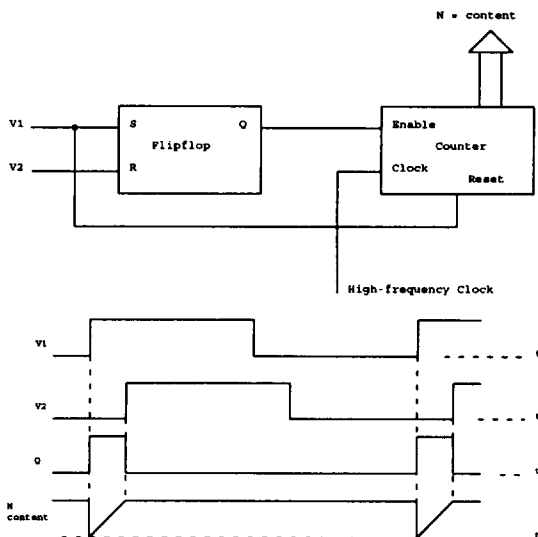
그림 1. Flip-Flop SPD

이고, 입력 필터의 대역폭 B의 대역 통과 잡음이 부과되어 있으면 샘플링 속도는 $f_s \cong 2B$가 된다.

Zero-Crossing SPD는 양의 영점 교차점과 양과 음의 영점 교차점 모두에서 샘플하는 방법의 두 가지가 있다. 양의 영점 교차점의 SPD는 입력 신호를 ADC(Analog to Digital Converter) 하는데 있어 국부 추정 신호를 클록으로 사용한다. ADC 출력은 위상차에 비례하는 출력을 발생한다. 이 출력은 ADC를 통과하여 천천히 변화하는 계단 함수이다.

Lead-Lag SPD는 양의 영점 교차 SPD의 경우와 유사하다. 그림2는 Lead/Lag SPD를 보여 준다. LL SPD는 국부기준에 의해 입력 신호를 샘플링 하여 +(Lead)와 -(Lag)의 두 경우의 2진법으로 단순히 출력한다. 보통 순차 필터를 사용하여 에러 전압을 고른다. 순차필터는 N before M filter혹은 Random Walk Filte를 사용한다.
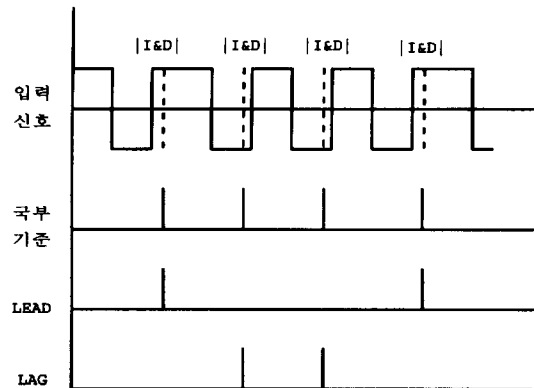
그림 2. Lead/Lag Sampling Phase Detector

3. SPD 알고리즘

본 논문에서 제안한 SPD 알고리즘을 설명하면 다음과 같다. 그림 3에서 NRZ Bit Input Stream 즉 NRZ의 입력 신호의 비트 구간은 Ti 이고 입력 신호의 천이를 주기적으로 샘플링하는 국부 기준 주기는 To이다. 여기서 To을 윈도(window)라 편의상 부르기로 한다. 윈도라 부르고자 하는 것은 LL SPD와 같이 국부 기준에 의해 입력 신호를 샘플링 하여 2진법으로 단순히 출력하는 것이 아니라 윈도 내 즉 To 구간 안에서 입력 신호의 천이 지점의 변화를 감지하는 영역이기 때문이다.
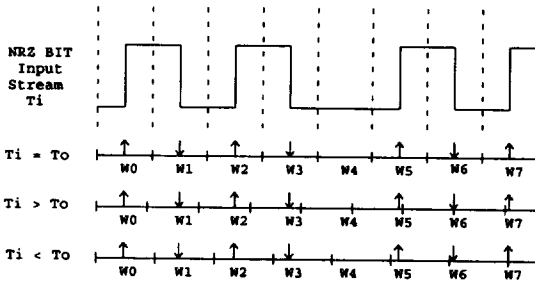
그림 3. 입력 신호의 비트 구간과 같은 윈도(Ti=To)와 작은 윈도(Ti<To) 및 큰 윈도(Ti<To)내에서의 입력 신호의 천이 지점

그림 3에서 입력 신호의 비트 구간 Ti와 윈도 To가 같은 크기라면(즉 Ti = To), 윈도 내 일정한 지점인 한곳에서만 입력 신호의 천이가 나타날 것이다. 그러나 입력 신호의 비트 구간에 비해서 다소 작은 구간의 윈도(Ti > To)에서는 입력 신호의 천이 지점이 윈도 내 일정한 지점에서 천이가 나타나지 않고 천이 지점이 우측으로 점차 이동되고 있음을 알 수 있다. 반대로 입력 신호의 비트 구간에 비해서 다소 큰 구간의 윈도(Ti < To)에서는 입력 신호의 천이 지점이 좌측으로 점차 이동되고 있음을 알 수 있다. 이것을 그림 4에서 보면 명확히 나타난다.

그림 4는 하나의 윈도 내에서 입력 신호의 천이 지점을 중첩하여 보았을 때의 그림이다. 그림에서 알 수 있는 것과 같이 윈도 내에서 입력 신호의 천이 지점이 우측 혹은 좌측으로 이동할 때는 각각 입력 신호의 비트 구간에 비해 윈도가 작거나 큰 것을 뜻한다.

입력 신호는 시간 축으로 관찰할 때 평균적으로 일정한 비트 구간을 갖지만 짧은 시간을 관찰할 때는 평균 구간보다 길거나 짧게 된다. 평균 비트 구간 즉 평균 주기에 대한 시간의 차이로 인한 위상의 변화를 위상 지터라 한다. 위상 지터로 인한 High 레벨의
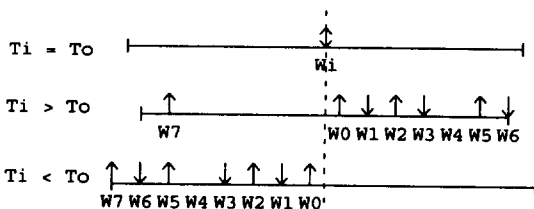
그림 4. 하나의 윈도 내에서의 중첩된 입력 신호의 천이 지점

비트 구간이 입력 신호의 평균 비트 구간보다 크거나 작을 경우에 대한 윈도 내의 입력신호의 천이 지점을 알아보자.

그림 5와 그림 6 그리고 그림 7과 그림 8은 각각 NRZ 입력 신호의 위상 지터가 있을 경우를 나타낸 것이다. 그림에서 보면 앞의 예와 달리 하나의 윈도 내에서 두 개의 입력 신호의 천이 지점이 나타난다. 따라서 입력 신호의 상승과 하강을 구분하여 각각의 윈도로 입력 신호의 천이 지점을 관찰해야 한다.

Ti = To인 윈도 내에서 입력 신호의 천이는 일정한 지점에 나타나지만 두개의 지점에서 천이가 나타난다. Ti > To일 때 입력신호의 상승과 하강 천이를 구분한 각각의 윈도에서 입력 신호의 천이 지점이 우측으로 이동한다. Ti < To일 때 입력 신호의 상승과 하강의 천이 지점을 구분하여 보면 각각 윈도 내에서 좌측으로 이동되는 것을 알 수 있다.

그리고 Low 레벨의 비트 구간이 입력 신호의 평균 비트 구간보다 작거나 클 경우에도 같은 결과를 얻을 수 있다.

따라서 이상의 예에서 알 수 있듯이 NRZ 입력 신호의 상승 천이와 하강 천이를 구분하여 각각의 윈도 내에서 천이 지점을 샘플링 하면 입력 신호의 비트
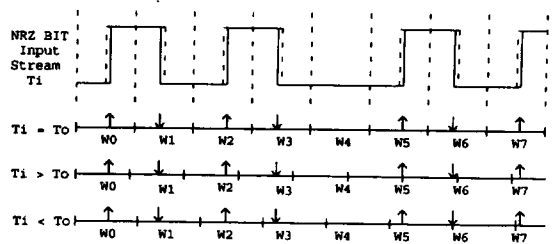
그림 5. High 레벨의 비트 구간이 평균 입력 비트 구간 보다 짧을 경우의 입력신호의 천이 지점
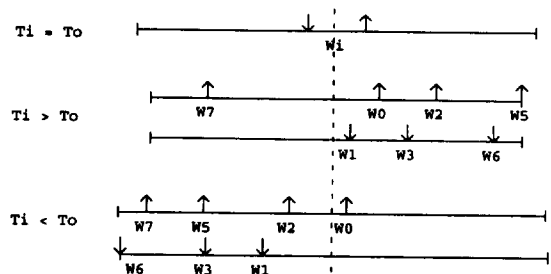
그림 6. High 레벨의 비트 구간이 평균 입력 비트 구간보다 짧을 경우의 중첩된 입력신호의 천이 지점
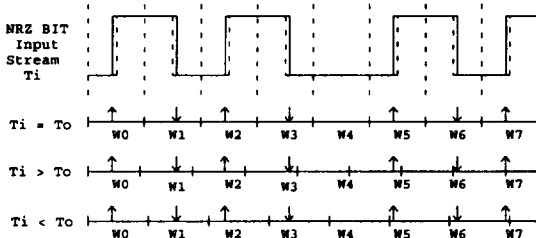
그림 7. High 레벨의 비트 구간이 평균 입력 비트 구간보다 클 경우의 입력신호의 천이 지점
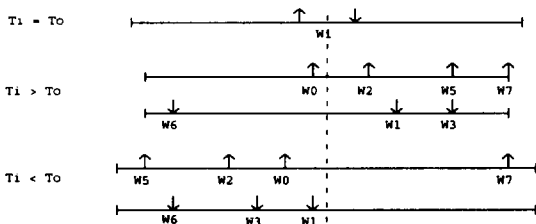
그림 8. High 레벨의 비트 구간이 평균 입력 비트 구간보다 클 경우의 중첩된 입력신호의 천이 지점

구간과 윈도와의 위상 차 즉 입력 주파수와 추정 주파수 사이의 주파수 차에 대한 출력을 얻을 수 있다. 이때 윈도 내에 천이 지점의 좌측 혹은 우측 움직임을 관측할 때 윈도는 링 버퍼 구조로 고려해야한다. 즉 윈도내의 좌우 양끝은 연결되어 있다고 고려한다. 이것을 알고리즘으로 정리하면 다음과 같다.

```
단계 1. window  To            : 초기 설정;
        window  To의 증감 량 : △T = 0;
        window  To 범위 내에서 상승 천이 지점
                   : Pe_Pointer 초기 설정 ;
        window  To 범위 내에서 하강 천이 지점
                   : Ne_Pointer 초기 설정;
        up_down_counter   = 0 ;

단계 2. window To 조정 : To = To + △T;

단계 3. window 주기 To내의  입력 신호의 천이 상태를 검사:
        if 입력 신호의 천이 =  상승 then
           goto 4 ;
        if 입력 신호의 천이  =  하강 then
           goto 5;
        end if
        goto 3;

단계 4. Pe_Pointer = window To 범위 내에서 입력 신호의 천이
지점;
        Pe_Turnig_Pointer = Pe_Pointer ;
        Pe_Pointer - Pe_Turnig_Pointer 하여 상태를 판별한 후
        if 천이 신호가 우측으로 이동 then
             goto 6 ;
        end if
        if  천이 신호가 좌측으로 이동 then
            goto 7;
        end if
        goto 3;

단계 5. Ne_Pointer= window To 범위 내에서 입력 신호의 천이
지점;
        Ne_Turnig_Pointer = Ne_Pointer ;
        Ne_Pointer - Ne_Turnig_Pointer 하여 상태를 판별한 후
        if 천이 신호가 우측으로 이동 then
           goto 6;
        end if
        if 천이 신호가 좌측으로 이동 then
           goto 7;
        end if
        goto 3;

단계 6. up_down_counter++;
       if up_down_counter >=  overflow then
             △T = +T_step;
                 up_down_counter = 0;
       else
             △T = 0;
      end if;
      goto 2;

단계 7. up_down_counter--;
      if up_down_counter <= underflow then
          △T = -T_step;
          up_down_counter = 0;
      else
          △T = 0;
     end if
     goto 2;
```

알고리즘 단계 2는 윈도 To가 입력 신호의 비트 구간을 동기 추적하는 과정이다. 여기서 △T는 위상 차를 보정하기 위한 출력이다. To는 참조 카운터(Ref_Period counter)로 구현한다. 여기서 up_down counter는 Random Walk Filter 필터이며 입력 신호의 비트 구간과 윈도사이의 위상 차에 따른 출력을 필터링 시킨다. 그림 9는 제안한 SPD 알고리즘을 구현한 블록도이다. 윈도 내의 입력 신호의 각 천이 지점은 최소 값인 0에서 최대 값인 Ref_Period counter의 값까지 될 수 있다. 이 값을 지정하기 위해 주기 카운터(Period_counter)를 사용하였다. Period_counter의 최대 값이 입력 신호의 비트 구간이 되도록 Ref_Period_counter를 조정한다.

NRZ 입력 신호인 Bit_Stream_input의 상승 천이 지점과 하강 천이 지점은 각각 Pe_pointer와 Ne_

pointer에 저장되고 이것은 각각 Pe_pointer와 Ne_pointer의 이전 값인 Pe_turning_pointer와 Ne_turn_pointer와 비교하여 Decision 로직에서 up과 down의 신호를 출력하고 up_down_counter에 의해 윈도 크기를 조정한다. 윈도 크기의 조정에 의해 윈도는 입력 비트 구간을 동기 추적한다.

제안된 SPD의 알고리즘를 사용하여 NRZ 랜덤 비트 동기 회로를 설계한다. SPD의 알고리즘으로 비트 동기 회로을 구현하려면 윈도에서 입력 신호의 샘플링 할 지점을 결정하여야 한다.
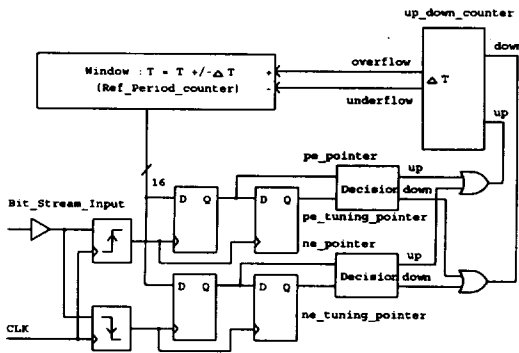
그림 9. 제안한 SPD 알고리즘을 구현한 블럭도

다음은 윈도 내에서 입력 신호의 샘플링 할 지점을 결정하는 알고리즘이다.

```
단계 1.
    초기 window 주기(ref_period_counter) 내에서 Sampling_point를 설정;
    Arith_Sampling_point=Sampling_point;
    Sampling_point의 증감량 : ΔS = 0 ;
    up_down_phase  = 0 ;
단계 2.
    Sampling_Point 조정 : Sampling_point <= Sampling_point + ΔS;
단계 3.
    if window 내의 입력 신호의 천이 없음 then
        goto 3;
    else if |Pe_Pointer - Ne_Pointer| < 허용 Duty 비 then
        Arith_Sampling_point=|Pe_Pointer-Ne_Pointer| / 2 ;
    else
        Arith_Sampling_point=(|Pe_Pointer-Ne_Pointer| + Ref_Period_Counter;
    end if;

단계 4.
  Sampling_point 와 Arith_Sampling_point 비교 한 후
    if Sampling_point  <  Arith_Sampling_point then
        goto 6;
    else
        goto 6;
    end if

단계 5 up_down_phase++;
    if up_down_phase = overflow then
        up_down_phase =0;
        ΔS = +S_step;
    else
        ΔS = 0;
    end if
    goto 2;

단계 6. up_down_phase--;
    if up_down_phase = underflow then
        ΔS = -S_step;
        up_down_phase;
    else
        ΔS = 0;
    end if
    goto 2;
```

여기서 Arith_Sampling_point는 입력 비트의 샘플링 지점을 연산한 값이고 Sampling_point는 실제 입력 비트를 샘플링하고 있는 지점이다. 샘플링 할 지점은 Period_counter의 전 구간의 값에서 하나를 결정하여야 한다. 샘플 지점은 다수 개로 결정하여 다수결 회로를 거쳐도 좋으나 여기서는 간단히 알고리즘의 구현을 위해 윈도내의 샘플 지점을 1개로 두었다.

샘플 지점(Sampling_point)은 단계 2에서 조정되며 이 샘플 지점은 단계 5와 단계 6에서 보는바와 같이 up_down_phase 카운터를 사용하여 Sampling_point의 증감량 ΔS를 구한다. up_down_phase 카운터는 Random Walk Filter 필터이다. Arith_Sampling_point는 단계 3에서 Pe_Pointer과 Ne_Pointer를 사용하여 연산한다. 여기서 허용 Duty 비란 시스템에 따라 달리 정할 수 있지만 Ref_Period_counter/2 로 두었다. 그림 10은 비트 샘플 지점을 결정하는 알고리즘을 구현한 시스템이다.

4. 실험 및 고찰

제안된 SPD의 알고리즘를 시뮬레이션 하였다. 그림 11은 NRZ 랜덤 비트 열에 대한 윈도 To와 up_down counter값의 관계를 알아보았다. 가로축은 윈도 축이며 세로축은 up_down counter값이다. 입력
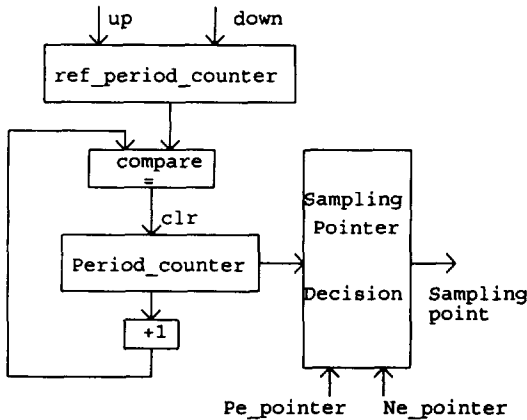
그림 10. 비트 샘플 지점을 결정하는 알고리즘의 블록도

비트 구간을 시스템 클록의 250배로 두고 윈도를 시스템 클록 1배에서 400배로 가변 하였을 때 up_down counter의 값을 조사하였다.

up_down counter는 초기에 '0'으로 설정되어 있고 입력 신호의 비트 구간과 윈도의 차이에 비례하는 음수와 양수의 값을 발생시킨다. up_down counter의 백분율은 입력 신호의 천이 개수에 대한 증가와 감소의 량을 나타낸다. 즉 up_down counter가 +100%로 표시한 것은 입력 신호의 천이 때마다 up_down counter가 증가된다는 것을 말하고 이것은 입력 신호의 천이마다 윈도 내 천이 지점이 우측으로 이동되는 것을 뜻한다. 그리고 up_down counter가 -100%로 표시한 것은 입력 신호의 천이마다 up_down counter가 감소된다는 것을 말하고 이것은 입력 신호의 천이마다 윈도 내 천이 지점이 좌측으로 이동되는 것을 뜻한다.

그림 11에서 보면 윈도가 시스템 클록의 250배에서 up_down counter의 값은 0%인 것을 알 수 있다. 이것은 윈도 값이 입력 신호의 비트 구간과 일치되어 있다는 것을 말한다. 윈도의 크기가 입력 비트 구간보다 큰 영역인 251부터 299 까지는 up_down counter값이 감소하며 윈도 주기가 설정된 입력 신호의 비트 구간보다 작은 영역인 215부터 249 까지는 up_down counter값이 증가한다. 특히 윈도 250을 중심으로 좌우 값이 +100%와 -100%인 것은 윈도의 크기가 입력 비트 구간과 조금 차이가 있을 때 윈도는 아주 빠르게 입력 신호의 비트 구간 250에 수렴할 수 있다는 것을 보여 준다. up_down counter가 증가
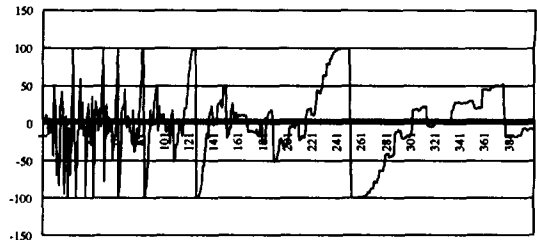
그림 11. NRZ 입력 신호 비트 구간을 시스템 클록의 25배로 설정하고 윈도를 가변 했을 때 up_down counter의 값의 백분율

하면 윈도가 크게 되어 입력 비트 구간에 수렴 할 수 있고 up_down counter가 감소가 하면 윈도가 작게 되어 윈도를 입력 비트 구간에 수렴할 수 있다. 그 밖의 영역의 윈도는 입력 비트 구간에 수렴하는 성질은 없지만 입력 비트 구간인 250에 대한 주파수의 배수인 윈도 125, 84, 62, 54에서는 작은 영역이지만 수렴하려는 현상을 보이고 있다. 그러나 250에 대한 주기의 배수를 가진 윈도에서는 수렴 현상은 나타나지 않는다. 이것은 그림 12의 시뮬레이션 결과에서 명확히 볼 수 있다.

그림 12는 입력 신호의 비트 구간을 40개의 시스템 클록으로 설정하여 윈도를 가변하여 본 것이다. 윈도 주기가 입력 신호와 일치하는 40에서 up_down counter의 값은 0%가 되어 증가와 감소가 있지 않다. 그러나 윈도의 주기가 입력 신호의 비트 구간보다 큰 영역인 41부터 48 까지는 up_down counter값이 감소한다. 그리고 윈도 주기가 입력 신호의 비트 구간보다 작은 영역인 35부터 39 까지는 up_down counter값이 증가한다. 따라서 외부 입력 신호의 비트 구간이 40개의 시스템 클록을 가지면 35부터 48 까지의 윈도의 주기를 가진 시스템이 윈도 40 주기로
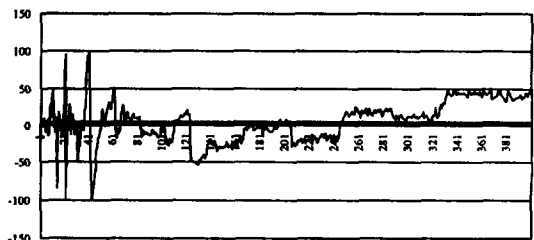
그림 12. NRZ 입력 신호 비트 구간을 시스템 클록의 40배로 설정하고 윈도를 가변 했을 때 up_down coun ter의 값의 백분율

수렴하게 할 수 있다. 40에 대한 주파수의 배수인 윈도 20에서는 작은 영역이지만 수렴하려는 현상을 보이고 있다. 그러나 20에 대한 주기의 배수인 윈도에서는 수렴 현상은 나타나지 않는다.

그림 11과 그림 12에서 보는 바와 같이 제안된 SPD 알고리즘을 적용하면 실험상으로 평균 윈도 의 +-10%이내의 입력 신호에 대해 주파수 동기 추적할 수 있음을 보여준다. 그리고 일단 주파수 동기 추적된 윈도는 다시 +-10%이내의 입력 신호에 대해 주파수 동기 추적할 수 있기 때문에 광범위한 추적 범위를 가진다.

제안된 SDP 알고리즘을 적용하기 위해서는 입력 신호의 비트 구간은 시스템 클록의 일정 배수 이상의 비트 구간을 가져야 한다. 그림 13을 보면 입력 신호의 비트 구간이 3개의 시스템 클록을 가질 때는 수렴현상은 윈도 주기 7을 중심으로 일어나는 것처럼 보인다. 따라서 실제 알고리즘의 적용을 위해서는 입력 신호의 비트 구간이 시스템 클록의 10배 이상이 되어야 한다. 그림 14에서 보면 윈도 주기 10에서는 수렴현상이 보이나 각각 최고 47과 -67의 백분율을 가지고 있어 주파수 동기 추적의 속도가 매우 느리다
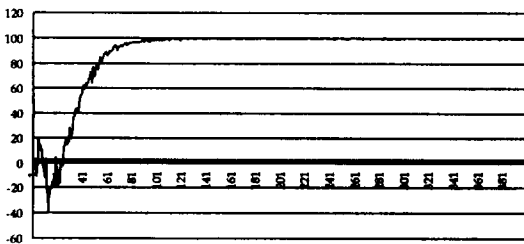
그림 13. NRZ 입력 신호 비트 구간을 시스템 클록의 3배로 설정하고 윈도를 가변 했을 때 up_down counter 의 값의 백분율
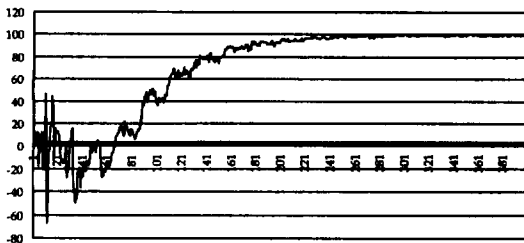
그림 14. NRZ 입력 신호 비트 구간을 시스템 클록의 10배로 설정하고 윈도를 가변 했을 때 up_down counter 의 값의 백분율

는 것을 알 수 있다.

디지털 랜덤 비트 동기회로에 제안된 SPD 알고리즘을 사용하여보았다. 디지털 랜덤 비트 동기회로를 VHDL(VHDL: Very High Speed Integrated Circuit Hardware Description Language)로 설계하고 PLD에 합성하였다. 사용된 툴은 Altera사의 MAX Plus II이고 합성에 사용된 PLD는 Altera사의 EPF10K10 QC208-4이다. 그림 15는 제안된 SPD를 사용한 비트 동기 회로의 VHDL 시뮬레이션 신호이다. 입력 신호는 bit_stream 이고 입력신호에 대한 비트 동기 출력은 sync_bit이다. 비트 동기 출력은 입력 신호의 비트 구간의 중간을 샘플링하는 펄스 신호이다. 비트 구간이 시스템 클록의 32배인 입력 신호에 대해서 윈도를 시스템 클록의 32배로 설정한 후 시뮬레이션을 하였다. 입력 신호의 시작점에서부터 40us 지점에서 입력 신호의 비트 구간을 32에서 34로 바꾸고 100us지점에서 다시 입력 신호의 비트 구간을 34에서 32로 되돌려 놓았을 때의 비트 동기 추적을 하는 것을 보여주고 있다. 여기서 제안된 알고리즘에서 사용하는 up_down_counter의 overflow와 underflow의 값은 입력 신호의 위상 지터와 같은 잡음 성분에 의존하여 설정한다. 그림15의 실험에서는 이 값을 +-4로 설정하였으므로 입력 신호의 비트 구간의 변화에 대해 동기 추적되어 윈도 (ref_period_counter)가 변하기까지 4개의 입력 신호의 천이가 필요하다.

따라서 40us 시점에서부터 8개의 입력 신호 천이가 있을 때 윈도 즉 ref_period_counter의 값이 2 감소되어 약 70us지점에서 윈도가 입력 비트 구간으로 수렴된다. 여기서 다시 입력 신호의 비트 구간을 34에서 32로 되돌려 놓으면 약 160us지점에서 윈도가 32로 되돌아간다.
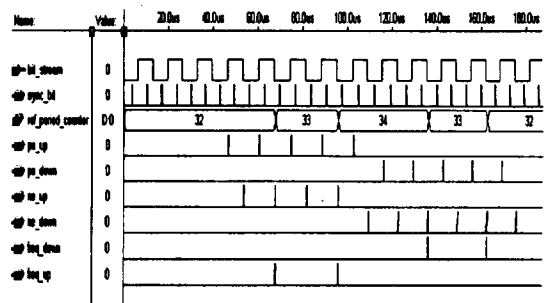
그림 15. 입력 신호의 비트 구간에 변화를 주었을 때 비트 동기 추적 시뮬레이션

그림 16은 EPF10K10QC208-4에 제안된 SPD 알고리즘을 합성한 후 외부 구형파 발생기에 의한 입력 신호에 대한 비트 동기 출력을 관찰하였다. 그림 17에서 위쪽의 구형 파는 함수 발생기에서 나오는 입력 신호이며 아래의 신호는 동기 추적된 출력 신호이다. 일단 동기화 되고 난 후에는 함수 발생기에서 나오는 입력 신호의 주파수를 서서히 변화시키더라도 동기가 유지된다. 그림 17은 NRZ 랜덤 입력 신호에 대한 비트 동기 추적을 관찰하였다. 동기된 출력 신호는 NRZ의 입력 비트 구간의 중앙을 정확히 샘플링한다. 이 실험에서도 랜덤 NRZ의 입력 비트 구간을 서서히 변화시키더라도 출력 신호는 입력 신호의 비트 동기를 유지함을 알 수 있었다.
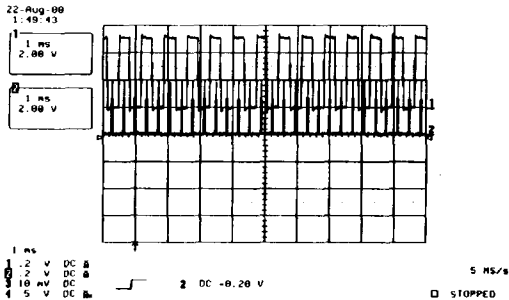
그림 16. 외부 구형파 발생기에 의한 입력 신호에 대한 비트 동기
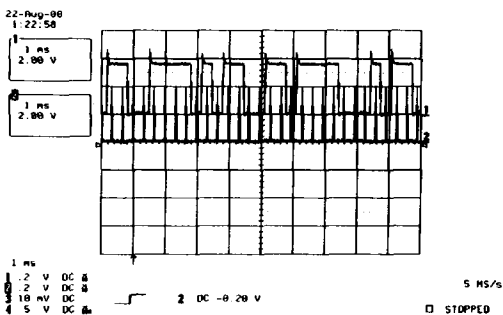
그림 17. NRZ 랜덤 입력 신호에 대한 비트 동기 추적

5. 결 론

본 논문에서는 데이터통신에 사용하는 NRZ 랜덤 비트 동기를 위한 새로운 방식의 표본위상검출기(SPD)를 제안하였다. 제안하는 SPD는 국부 기준 신호의 주기에 해당하는 윈도에서 입력 신호의 천이 지점의 변화를 감지하여 위상 차를 출력한다. 제안된 SPD를 NRZ 랜덤 비트 동기 회로에 적용해 본 결과 효과적임을 알 수 있었다.

제안하는 SPD는 LL SPD의 범주에 속할 수는 있지만 LL SPD의 동작 원리와는 다르다. LL SPD는 국부기준에 대한 RZ 입력 신호의 sampling에 의해서 단순히 Lead-Lag의 2진법으로 동작한다. 그러나 제안된 SPD는 입력신호로 NRZ를 대상으로 할 수 있을 뿐 아니라 국부 기준의 주기에 해당하는 윈도 내에서 입력 신호의 천이 지점의 변화에 의해 동작된다는 점에서 근본적 원리를 달리한다.

제안된 SPD 알고리즘을 적용하면 실험상으로 입력 신호의 비트 구간에 대해 평균 +-10%이내의 윈도는 동기 추적되는 것을 알았다. 그리고 일단 동기 추적된 윈도는 다시 +-10%이내의 입력 신호에 대해 동기 추적할 수 있으므로 광범위한 추적 범위를 가진다.

제안된 SDP 알고리즘을 적용하기 위해서는 입력 신호의 비트 구간은 시스템 클록의 일정 배수 이상의 비트 구간을 가져야 한다.

지금까지 새로운 방식의 SPD을 제안하였고 제안된 SPD의 타당성을 검증하기 위한 시뮬레이션 해석을 하였다. 그리고 제안된 SPD를 비트 동기 회로에 적용한 결과 NRZ 랜덤 비트 동기에 적합하다는 것을 실험적으로 밝혔다. 차후의 과제는 제안된 SPD에 대한 수학적 해석과 지터 등의 잡음에 대한 안정성 검증이다.

참 고 문 헌

[1] William C. Lindsey and Chak Ming Chie, "A Survey of Digital Phase Locked Loop", Proceeding of Ieee, Vol. 69, pp. 410-431, 1981

[2] Jack K.Holmes, "Performance of a First-Order Transition Sampling Digital Phase-Locked Loop Using Random-Walk Models", IEEE Trans. on Commun, Vol. Com-20, pp. 119-131, March 1966.

[3] Flloyd M. Gardner. Phase Lock Techniques, John Wiley&Sons. Inc., 1979.

[4] James R. Cessna, and Donald M.Levy, "Phase Noise and Transient Times for a Binary Quan-

tized Digital Phase-Locked Loop in White gaussian Noise", IEEE Trans. on Commun. Vol. Com-20, No. 2, pp. 94-164, Apr.1972.

[5] Hisao Yamamoto and Shinsaku Mori, "Performance of a Binary Quantized All Digital Phase-Looked Loop with a New Class of Sequential Filter, IEEE Trans. on Commun. Vol. Com-26, No. 1, pp. 35-45. Jan. 1978.

[6] Donald G. Troha " Digital Phase-Locked Loop Design using SN54/74297", Texas Instruments Application Report, 1982.

[7] John C. Y. Huang, Kamilo Feher, Michel Gendron, "Techniques to generate ISI and Jitter Free Band limited Nyquist Signals and a Method to Analyze Jitter Effects, IEEE Trans. on Commun. Com-27, No. 11, pp. 1700-1711, Nov. 1979.

[8] Engel Rosa "Analysis of Phase-Looked Timing Extraction Circuits for Pulse Code Transmission", IEEE Trans. on Commun. Vol. Com-22, No. 9, pp. 1236-1249, Sept. 1974.

[9] 최형진, "동기방식 디지털 통신", 교학사, 1995.

박 세 현

1980년 경북 대학교 공과대학 전자공학과 학사
1982년 경북 대학교 대학원 전자공학과 석사
1985년 아주 대학교 대학원 전자공학과 박사
1992년~현재 국립안동대학교 전자정보산업학부
1997년 11월 국민 포장 수여
1997년~1999년 국립 안동대학교 공과대학 학장
1999년~2000년 미시건 주립대학 전기 컴퓨터 공학과의 겸임 교수
관심분야 : 디지털시스템 설계, 컴퓨터구조, 유전자알고리즘

박 세 훈

1992년 Arizona State University 전기공학 PhD.
1990년 Arizona State University 전기공학 석사
1980년 경북대학교 전자공학과 학사
1987년 현대전자 반도체 사업부
1993년 한국전자통신연구소(ETRI)
1995년 안동대학교 전자정보산업학부
관심분야 : ASIC 설계